Inf Retrieval (2010) 13:534–567
DOI 10.1007/s10791-010-9135-7

Why finding entities in Wikipedia is difficult, sometimes

Gianluca Demartini · Claudiu S. Firan · Tereza Iofciu · Ralf Krestel · Wolfgang Nejdl

Received: 1 May 2009 / Accepted: 19 April 2010 / Published online: 18 May 2010

Abstract Entity Retrieval (ER)—in comparison to classical search—aims at finding individual entities instead of relevant documents. Finding a list of entities requires therefore techniques different to classical search engines. In this paper, we present a model to describe entities more formally and how an ER system can be build on top of it. We compare different approaches designed for finding entities in Wikipedia and report on results using standard test collections. An analysis of entity-centric queries reveals different aspects and problems related to ER and shows limitations of current systems performing ER with Wikipedia. It also indicates which approaches are suitable for which kinds of queries.

Keywords Entity search · Evaluation · Model · Algorithms · Experimentation

1 Introduction

1.1 Motivation

Finding entities on the Web is a new search task which goes beyond the classic document search. While for informational search tasks (see Broder 2002) for a classification) document search can give satisfying results for the user, different approaches should be

G. Demartini · C. S. Firan (✉) · T. Iofciu · R. Krestel · W. Nejdl
L3S Research Center, Leibniz Universität Hannover, Appelstrasse 9a, 30167 Hannover, Germany
e-mail: firan@L3S.de

G. Demartini
e-mail: demartini@L3S.de

T. Iofciu
e-mail: iofciu@L3S.de

R. Krestel
e-mail: krestel@L3S.de

W. Nejdl
e-mail: nejdl@L3S.de

followed when the user is looking for specific entities. For example, when the user wants to find a list of “European female politicians” it is easy for a classical search engine to return documents about politics in Europe. It is left to the user to extract the information about the requested entities from the provided results. Our goal is to develop a system that can find entities and not just documents on the Web.

Being able to find entities on the Web can become a new important feature of current search engines. It can allow users to find more than just Web pages, but also people, phone numbers, books, movies, cars, etc. Searching for entities in a collection of documents is not an easy task. Currently, we can see the Web as a set of interlinked pages of different types, e.g., describing tasks, answering questions or describing people. Therefore, in order to find entities, it is necessary to do a preprocessing step of identifying entities in the documents. Moreover, we need to build descriptions of those entities to enable search engines to rank and find them given a user query. Applying classical Information Retrieval (IR) methodologies for finding entities can lead to low effectiveness as seen in previous approaches (Bast et al. 2007; Cheng et al. 2007; Pehcevski et al. 2008). This is because Entity Retrieval (ER), that is, finding entities relevant to a query, is a task different than document search. An example of an ER query is “Airports in Germany” where a relevant result is, e.g., “Frankfurt-Hahn Airport”. Airports not in Germany or entities other than airports would not be relevant to the given query. It is crucial to rely on consolidated information extraction technologies if we do not want to start with an already high error that the ranking algorithms can only increase.

1.2 Entity retrieval related tasks

With the current size of the Web and the variety of data it contains, traditional search engines are restricted to simple information needs. Complex queries need, usually, a lot of effort on the user side in order to be satisfied. We can observe different search tasks related to this scenario:

Entity retrieval. Finding entities of different types is a challenging search task which goes beyond classic document retrieval as well as beyond single-type entity retrieval such as, for example, the popular task of expert finding (Bailey et al. 2007). The motivation for the ER task is that many user queries are not looking for documents to learn about a topic, but really seek a list of specific entities: countries, actors, songs, etc. Examples of such informational needs include ‘Formula 1 drivers that won the Monaco Grand Prix’, ‘Female singer and songwriter born in Canada’, ‘Swiss cantons where they speak German’, and ‘Coldplay band members’. The query ‘countries where I can pay in Euro’ is answered by current web search engines with a list of pages on the topic ‘Euro zone’, or ways to pay in Euros, but not with a list of country names as the user is asking for. Note that while a single query refers to a single entity type, a system must be able to answer queries for different entity types (differently from an expert search system where the response is always of type person). A commercial prototype performing this task is Google Squared.[1]

Question answering. It must also be mentioned how Entity Retrieval task relates with Question Answering (QA). Common queries in the QA context usually are of type Who, When, Where, Why, How Many. That is, they expect a precise answer as, for example, a number or a name instead of a list of entities. ER queries have considerable similarities

[1] http://www.google.com/squared.

with QA “list” questions where the user is looking for a list of items as a result (e.g., “What companies has AARP endorsed?”). In the evaluation benchmarks, QA queries usually consist of sets of questions about a particular topic: this might let the system approach the problem in a different way, e.g., by mining documents retrieved with a keyword query or by exploiting the answer of previous questions on the same topic (e.g., “What does AARP stand for?”). In conclusion, there are similarities between ER and QA queries. In particular for list QA queries we can imagine ER technologies described in this paper exploited, among other things, by QA systems to perform better on this particular type of queries.

Related entities. Another related task is finding entities similar or related to other entities. In this case the user might have in mind a search query consisting of an example entity. For a given entity, such as “New York”, one would expect to find as associated entities places to visit in New York (e.g., “Empire State Building”, “Statue of Liberty”), connected historical events (e.g., “September 11, 2001”) or famous people (e.g., “Rudy Giuliani”), etc. in a faceted-search fashion. The associated entities can be presented to the user as a lists or grouped by type and other properties (e.g., date). For a query “Albert Einstein”, the system may return related entities like, for example, “Germany”, “Nobel prize”, “physics”, “Lieserl Einstein”, etc. This task is different from ER as the result set may contain entities of different types. Here the system provides the user with a browsing opportunity rather than with a list of retrieved entities as for ER. A commercial prototype performing this task is Yahoo! Correlator.[2]

1.3 Our contribution

In this paper we focus on the ER task and we first propose a general model for finding entities of a given type and we show how this can be applied to different entity search scenarios. We generalize this search task and identify its main actors so that we can optimize solutions for different search contexts such as, for example, the Wikipedia corpus. We also present results for list completion when starting from given entities. Building on top of the designed model, we developed search algorithms based on Link Analysis, Natural Language Processing (NLP), and Named Entity Recognition (NER) for finding entities in the Wikipedia corpus. Moreover, we experimentally evaluate the developed techniques using a standard testbed for ER. We show that these algorithms improve significantly over the baseline and that the proposed approaches—incorporating Link Analysis, NLP and NER methods—can be beneficially used for ER in Wikipedia. We evaluated our algorithms for entity ranking only on Wikipedia as they are designed for this specific context and can not be directly applied to the Web at large. It will be a future step to extend the approach to the entire Web of Entities. We also perform an analysis of which ER queries are difficult and which are easy for state-of-the-art systems.

The main contributions of this paper are thus a formal entity ranking model along with a collection of tested methods designed for the Wikipedia scenario. Additionally, we present a per-topic analysis of the ER systems performance and an analysis of the different kinds of entity queries.

The paper is structured as follows. We first start by defining a model for ER (entities, queries, and the ER system) in Sect. 2. Then, in Sect. 3 we present an ER system we use in our experiments including a standard benchmark data set. In Sects. 4 and 5 we depict all

[2] http://www.correlator.sandbox.yahoo.net.

the different approaches we use and their effectiveness evaluations. Section 6 analyzes the queries and discusses the results. Related work is presented in Sect. 7. We finally conclude the paper and present future improvements in Sect. 8.

2 A formal model for entity ranking

Searching for information on the Web is a very common task that many search engines deal with. The difficult part is to distinguish if the answer to the user's information need is just a fact that appears in different pages or if it is information about a specific object, an entity. Searching for named entities, such as "the first dog on the Moon" or general entities like "dog species bred in England" is quite different than searching for "tips and tricks on raising dogs" (i.e., informational queries).

The problem of ranking entities in IR can be split in several steps. First, the user's information need has to be translated into a query which has to be interpreted and the *entity need* has to be extracted. The search engine has to understand what type of entity the user is searching for and what properties the retrieved entities should have. In the next step, relevant results are retrieved. The results have to be retrieved according to the entity description which can include different properties, e.g., the type. We propose in the following section a model for the entire ER process to help with the understanding of the problem and of the general ER flow. This model can be instantiated in a number of different contexts such as, for example, Wikipedia.

2.1 Entities

The central part of the model is the set of entities. An entity e^i is something that has separate and distinct existence and objective or conceptual reality.[3] An entity is represented by its unique identifier, and by a set of properties described as (<*attribute*>, <*value*>) pairs (see Fig. 1). The properties of an entity can include, for example, its name or its type. Moreover, it is important to notice that relations can be present between entities. It is possible to model these relations as other properties using (<*attribute*>, <*value*>) pairs where the value would be the target entity of the relation. This representation of relations is consistent with previous work on entity relation search (Zhu et al. 2008).

We can now define the entity description $d(e^i) = \{ID^i, P^i\}$ for the entity e^i as composed of an entity identifier $ID^i = id(e^i)$ and a set of properties $P^i = \{(a^i{}_1, v^i{}_1)\ldots(a^i{}_n, v^i{}_n)\}$ of the type (<*attribute*>, <*value*>) pairs. For example, the soccer player "Alexandre Pato" could have as ID the automatically generated unique identifier *ap*12*dH*5*a* and properties such as (*born_in*, 1989) or relations with other entities such as (*playing_with*, *acm*15*hDJ*) where *acm*15*hDJ* is the ID of the soccer club "A.C. Milan".

2.2 Data sources

In order to create the entity descriptions $d(e^i)$ (see Sect. 2.1) we need to extract data about entities from several sources. For example, for describing the company "Microsoft Corporation" we might want to harvest the Web in order to find all the facts and opinions

[3] Clearly, it is not easy to define what an entity is and there are many studies in philosophy trying to define it. To keep the problem simple we consider retrievable entities all *objects/things* about which the Web contains information.

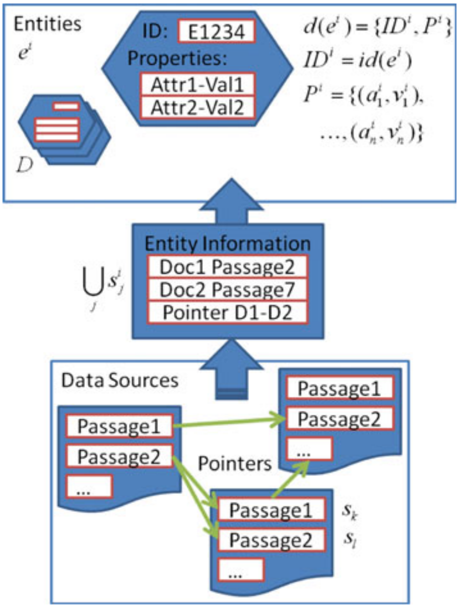


Fig. 1 Entities and their extraction from different data sources

about this entity. For this reason, we call *data sources* the provenance of the information we collect in an entity description. We define a data source s_j as any passage of a digital document. This can be an XML element, a paragraph of an e-mail, a blog post on the Web, etc. Each data source s_j can be about one or more entities. The aggregation of all the data sources about the same entity e^i (noted as $\bigcup s_j^i$) will create the properties part P^i of the entity description $d(e^i)$ as defined in Sect. 2.1. This would define inferring the description of an entity as: $\bigcup_j s_j^i \Longrightarrow P^i$. The relations between entities are also inferred from the data sources and are part of P^i.

2.3 Users' information need

After modeling the entities and the data sources used for creating their description, we want to model a user searching for entities. We assume that a user has an information need, that is, she wants to find a list of entities that satisfy some properties. It is a user task to create, starting from the information need, a query, either using keywords or natural language questions, that can be processed by the system. The user query will describe the set of properties that an entity should satisfy for being relevant. For example, a query might indicate the type of entities to be retrieved (e.g., "cars") and distinctive features (e.g., "German", "hybrid"). A real world example is given by the search engine http://www.sindice.com where the user can issue queries like "Washington class:person" specifying the type of results she wants to get. A query q is defined, similar to the entity descriptions, as a list of (<*attribute*>, <*value*>) pairs. Thus, $q = \{(a_1, v_1)\ldots (a_n, v_n)\}$.

2.4 Entity ranking system

At this point, a collection of entity descriptions $D = \{d(e^1)\ldots d(e^n)\}$ and a user query q is available. An Entity Ranking System (ERS) will now take as input these two elements and will return a ranked list of entities $E = \{e^i\ldots e^j\}$ (Fig. 2 shows a sketch of the flow inside

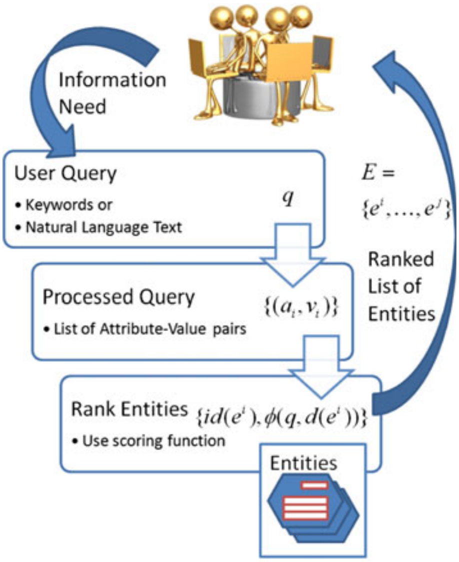


Fig. 2 The entity ranking flow

the ERS). In order to do this, an ERS will hard-code a *scoring function* $\phi(q, d(e^i))$ that returns a score (i.e., a real number) for a given user query q and entity description $d(e^i)$. This score represents the confidence or probability of the entity e^i of being relevant to the query q. In this way the ERS will be able to rank the entire set of entities according to the confidence of being relevant to the user query. Of course, the scoring function can take into account several evidences of relevance such as the comparison between properties in q and properties in $d(e^i)$, the popularity value of the entities in the collection (e.g., PageRank), or give more importance to a particular property (e.g., the type of entities to be returned).

2.5 Application scenarios for ER

As an initialization step, it is necessary to assign a global identifier for each entity in the collection. Attempts to generate global unique identifiers are already underway, e.g., the OKKAM[4] European Integrated Project is dealing with ID generation on the Web. One simple application scenario would be ranking consumer products (i.e., entities) where a customer provides as query a list of constraints (e.g., brand, color, size, etc.). ER can be also performed on the Web, where the definition of an entity is not as trivial as in the enterprise example. The entity description will then contain attributes of the entities mentioned in sentences of several Web pages referring to the entity. Relations between entities can then be constructed from links between Web pages as well as references between sentences or paragraphs.

Another application scenario which keeps the main information as in the Web application scenario but also adds some structure is the Wikipedia model for ER. In this case we consider in D any entity e^i that has its own page in Wikipedia. With this assumption we can easily see these pages as the entity description $d(e^i)$ and the set of the Wikipedia pages that describe an entity as the collection D. Of course, in Wikipedia there are pages which do not describe a

[4] http://www.fp7.okkam.org/.

particular entity as, for example, the "List of…" pages. The challenge is to identify which are not entity pages and discard them from D. For each entity the (<*attribute*>, <*value*>) pairs can be build, for example, out of the info-boxes of the Wikipedia pages which contain factual information about the described entity (for example, articles about people contain information about name, birth date, birth place, etc.). In the Wikipedia scenario the sources of information are the people and each s^i_j contributing to $d(e^i)$ can be reconstructed from the edit history of each page allowing also to associate trust values in order to weight more particular sources (see also Adler and de Alfaro 2007 about such computation). For defining the *type* property in $d(e^i)$ the Wikipedia category information can be used. Relations between entities can be discovered analysing the Wikipedia internal links between pages. The query can be built by the user providing some keywords describing interesting properties plus the selection of a Wikipedia category in order to provide information about the type of entities which are requested. The ranking function $\phi(q, d(e^i))$ should use both information about the properties and the type in order to produce the best ranking.

The specific Wikipedia scenario is slightly different from the general Web scenario as Wikipedia is more clearly structured. It is easy to define an entity as having its own Wikipedia page (i.e., each Wiki page is about one entity)—in the general Web scenario we would have to segment Web pages to extract only sections related to the entity and discard other parts like advertisements or navigational headers. Moreover, it is also easy to extract the entity type from a Wikipedia page, as one of the entity attributes $d(e^i)$, by just considering the Wikipedia categories the page belongs to—the Web scenario would require a thorough NLP processing of the text in order to find phrases describing the entity (e.g., "Mexico *is a* country"). We also make use of the YAGO ontology (see Sect. 4.1) which is built from Wikipedia and WordNet. If the same system architecture were to be applied to the Web, a new ontology would have to be built in order to make the results comparable. YAGO is also being used in other scenarios than Wikipedia: http://www.Revyu.com (Heath and Motta 2008) uses Yago class definition in order to assign types to the objects of reviews; in Raimond et al. (2008) the authors use links between DBpedia and YAGO for interlinking singers, songs, music events, etc. in music datasets. Finally, there is much more content to be found on the Web, while Wikipedia only focuses on some, more common, topics and entities (e.g., we can not find members of a particular organization only from Wikipedia). Nevertheless, Wikipedia is a very good starting point for the emerging task of entity retrieval, and we will focus on the Wikipedia scenario in the remainder of the paper. Other algorithms might be developed for ER on the Web still following the proposed model.

3 A baseline system for finding entities in Wikipedia

In this section we describe INEX and Wikipedia and present a baseline system for Entity Retrieval in Wikipedia.

3.1 INEX and the Wikipedia collection

In 2007, the evaluation initiative INEX[5] has started the XML Entity Ranking track (INEX-XER) to provide a forum where researchers may compare and evaluate techniques for systems that retrieve lists of entities (Vries et al. 2007). In the INEX initiative a snapshot of the English Wikipedia (Denoyer and Gallinari 2006) was used as evaluation

[5] http://www.inex.otago.ac.nz/.

corpus. The assumptions are that each entity described in Wikipedia is a possible candidate and the respective page represent the real entity to be retrieved. Additionally, the relevance is defined as binary.

Entity Retrieval can be characterized as 'typed search'. In the specific case of the INEX-XER track, categories assigned to Wikipedia articles are used to define the *entity type* of the results to be retrieved. Topics are composed of a set of keywords, the entity type(s), and, for the list completion task, a set of relevant entity examples. The two search tasks evaluated in the context of INEX-XER are entity ranking (XER) and entity list completion (LC). We will focus on entity ranking where the goal is to evaluate how well systems can rank entities in response to a query; the set of entities to be ranked is assumed to be loosely defined by generic categories, given in the query itself, or by some example entities respectively. In the entity list completion task, the categories from the queries are not used, but a set of example entities provided for each topic. Thus from the example entities the system has to learn relevant information to describe the retrieved entities, such as category information and link information. For completeness, we will also present results on the LC task in order to show how our system can be adapted to the alternative LC setting.

The document collection used for evaluating our approaches is the Wikipedia XML Corpus based on an XML-ified version of the English Wikipedia in early 2006 (Denoyer and Gallinari 2006). The considered Wikipedia collection contains 659,338 Wikipedia articles. On average an article contains 161 XML nodes, where the average depth of a node in the XML tree of the document is 6.72. The original Wiki syntax has been converted into XML, using general tags of the layout structure (like *article, section, paragraph, title, list*, and *item*), typographical tags (like *bold, emphasized*), and frequently occurring link-tags. For details see Denoyer and Gallinari (2006).

The official evaluation measure used at INEX-XER 2007 was Mean Average Precision (MAP) aiming at evaluating the overall entity ranking produced by the systems. In 2008, a new evaluation measure was introduced. INEX-XER 2008 used as official metrics xInfAP (Yilmaz et al. 2008). The metrics xInfAP is an estimation of Average Precision (AP) for the case where the judgement pool has been built with a stratified sampling approach. This means that the complete collection of documents is divided into disjoint contiguous subsets (strata) and then documents are randomly selected (sampling) from each stratum for relevance judgement. In this case it is possible to give more importance to documents retrieved higher by ER systems (e.g., by having a complete assessment of top 20 retrieved results) still going down into the list of retrieved entities (e.g., by having a partial assessment of results retrieved between rank 30 and 100). The metrics xInfAP is computed exploiting (similarly to infAP (Yilmaz et al. 2006)) the estimation of Precision at each relevant documents in each stratum. In the following, we report values for xInfAP as it was used as official INEX-XER 2008 evaluation metrics and we use the INEX-XER 2008 testbed as it consists ER topics created for this purpose and assessed by participants. Even if it was shown to be redundant (Webber et al. 2008), we additionally report Precision at 10 retrieved entities (P@10) as it may be more intuitive for the reader to understand how well the system performs. The official set contains 35 ER designed topics. An example topic can be seen in Table 1.

Although this task seems easy given the Wikipedia corpus, we have to retrieve results matching the sought type (i.e., the type resulting from the entire information need, not necessarily the assigned Wikipedia category). Relevant results for the example given in Table 1 ("National capitals situated on islands") would thus be: "London", "Tokyo", or "Jakarta", all of these being capitals situated on islands. Irrelevant results, although they still contain some information related to the Topic, would be the tourist attraction "London Eye" (which is situated in "London"), or even "List of capitals" (the page listing known capitals).

Table 1 INEX-XER 2008 entity ranking topic example

Topic ID	#109
Title	National capitals situated on islands
Description	I want a list of national capitals that are situated on islands
Narrative	Each answer should be the article about a nation-level capital whose geographic area consists of one or several islands
Category	(10481) Capitals
Examples	London, Tokyo, Jakarta

3.2 Processing pipeline

The processing pipeline is presented in Fig. 3. The first step is the creation of the inverted index from the XML Wikipedia document collection. We use standard retrieval techniques as initial step in order to focus on the ER-specific aspects of the approach that can improve the effectiveness of the system.

Starting from the raw structured XML documents, we created an inverted index using TFxIDF weighting scheme and cosine similarity as ranking function.[6] We indexed separately each article element (e.g., *title*, *text*, *textstem*, *categories*, *links*, etc.) so that we can perform the search on the different fields of a Wikipedia entity. The main advantage of such an approach is that we can search with different queries the content of the article and the category information. We can also exclude the title from the search as it usually does not contain discriminative information for ER queries. Important is also the anchor text of outgoing links in a page which usually describes related entities. For example, the Wikipedia page of Albert Einstein links to the page “Nobel Prize in Physics” using this same string as anchor text. By looking at this anchor text we can infer that the entity “Nobel Prize in Physics” is related to “Albert Einstein” which can help us in answering queries like, e.g., “Nobel prize winners”.

After the creation of the index, the system can process the INEX-XER 2008 topics. Different approaches are adopted for building queries out of INEX Topics. Four modules are used interchangeably or complementary as main resources for our algorithms (see Fig. 3):

1. *Wikipedia Taxonomy* is used for getting Wiki Category Links (see Sect. 4.2),
2. *Entity Linkage Information* is needed for exploring outgoing Links of Wikipedia entities (see Sect. 4.2),
3. the *NLP Processing* is used to find lexical compounds, synonyms and related words, named entities and filter information in the query (see Sect. 5), and
4. the *YAGO* ontology is used as underlying knowledge base (see Sect. 4.1).

The INEX Topic is processed using these modules in order to create a disjunctive query starting from Title information, along with the specified Category from the Topic. For the XER task we only use the Title and Category information as the Description part of the topic contains complete sentences and it is realistic to assume that users would not post such long queries to a search engine. Moreover, the Narrative part of the topic is intended only as assessment guidelines. For the LC task we use the Title and the Example information from the topic. In this situation the desired category is learned from the given

[6] We used the Lucene tool with default configuration. http://www.lucene.apache.org.

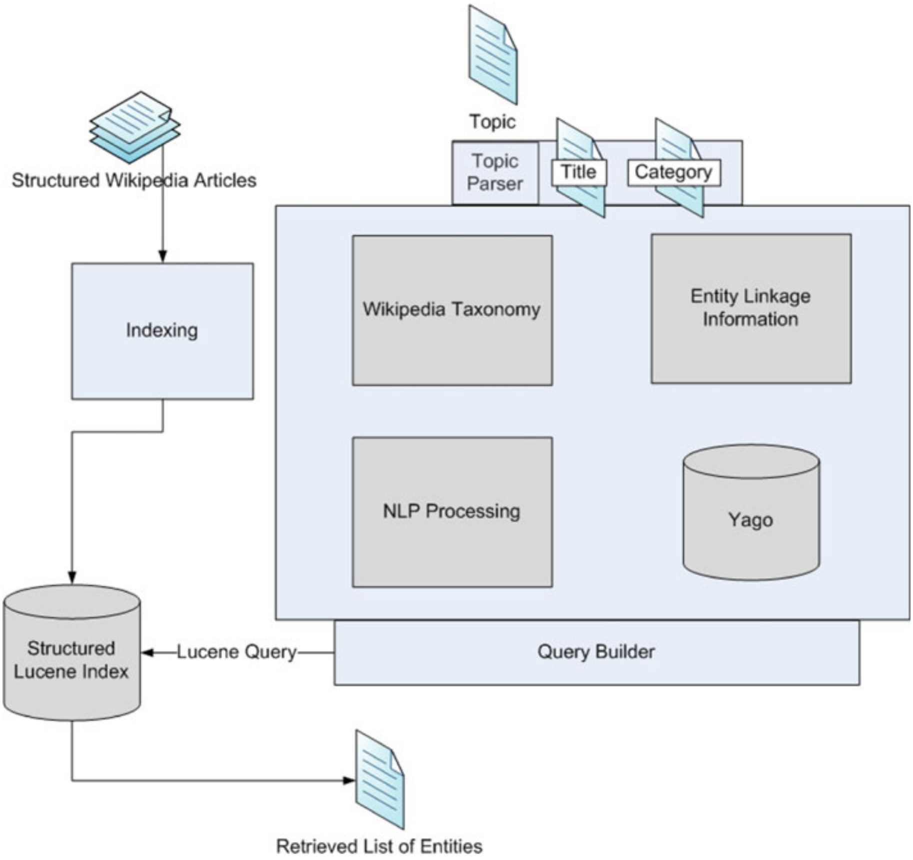


Fig. 3 Processing pipeline for the entity ranking system

example entities. After the generation of the query, the index can be queried and a ranked list of retrieved entities is generated as output.

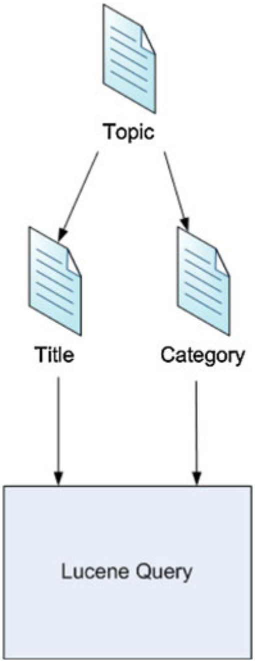
In all our experiments we use standard stemming and stopword removal techniques on the Wikipedia article content, while, when searching the category information, we do not apply a stemming algorithm in order to have a perfect match on the category information. Other combinations with regard to stopwords and stemming did not prove so effective. Moreover, we remove all results which have a title starting with "List of" as we do not expect such articles to be relevant entities to any ER query.

3.3 Baseline approach and notations

We use the following notations for describing the algorithms presented throughout this article:

- $W^T = \{w_1^T, \ldots, w_n^T\}$—the words in the given Topic Title;
- $W^C = \{w_1^C, \ldots, w_n^C\}$—the words in the given Topic Category;
- $W^{LC} = \{w_1^{LC}, \ldots, w_n^{LC}\}$—the words in the categories from the given Example Entities in the Topic.

Fig. 4 Query creation using only topic information



- $W^T_{Adj} = \{w^T_{Adj_1}, \ldots, w^T_{Adj_n}\}$—the adjectives in the Topic Title;
- $W^T_{Noun} = \{w^T_{Noun_1}, \ldots, w^T_{Noun_n}\}$—the nouns in the Topic Title;

As a baseline approach for constructing the query, we consider only the information given in the title part of the topic, as presented in Fig. 4. For search we use the Vector Space Model and ranking is done using standard cosine similarity and TFxIDF weighting scheme.[7] We construct a disjunctive query containing both textual and contextual information (i.e., keywords and category information). For the textual part of the query we consider the keywords from the title of the topic which we run against *TextStem* field (which contains the main textual area of a Wikipedia page, with stemmed terms) in the index. In the contextual part of the query we consider the category information from the topic which we run against the *Categories* field (containing the Wikipedia categories as listed on the Wikipedia pages).

The query part searched in the Wiki page text will thus contain following terms:

$$w_i \in W^T$$

For example, for the topic described in Table 1, the query resulting after stopword removal and stemming is the following:

text:(nation capit situat island)
category:(capitals)

[7] All search and ranking settings were left as default in Lucene.

In the case of the List Completion task we extract the categories from the given example entities, as presented in Fig. 5. Each topic has between three to five example entities and we consider the categories that at least two example entities belong to, where applicable. For the topics where there are no categories with two common entities we move the threshold down. For example, for the topic in Table 1, the three example entities belong to the following categories:

- Categories with two common entities—capitals in asia, host cities of the summer olympic games, coastal cities;
- Categories with one entity—capitals in europe, cities in england, london, kanto region, tokyo, destroyed cities, harbours, visitor attraction in london cities in indonesia, provinces of indonesia, london squares, jakarta, london parks and commons.

As there are categories with two common example entities we use only those and the resulting LC query is the following:

text:(nation capit situat island)
category:(capitals in asia, host cities summer olympic games, coastal cities)

In the following sections we present two groups of approaches for improving effectiveness of ER in the Wikipedia context. In Sect. 4 we describe how ER effectiveness can be improved by extending the Wikipedia category information with external knowledge. In Sect. 5 we present approaches for refining the user query using IE and NLP techniques.

4 Structure based techniques for entity retrieval

One of the main issues in performing the ER task on Wikipedia is the incompleteness of the Wikipedia category structure. Relevant results may belong to different categories than the ones provided by the user. In this section we first define algorithms aiming at improving the category information available by means of a highly accurate ontology build on top of WordNet and Wikipedia (Sect. 4.1). After this, we focus on how to better understand the user provided keyword query (Sect. 5) as a different approach for improving the effectiveness of the ER task (Demartini et al. 2008).

4.1 Category refinement by means of a highly accurate ontology

The lack and the imprecision of category information in Wikipedia can be attenuated by using Ontologies to identify relevant categories.

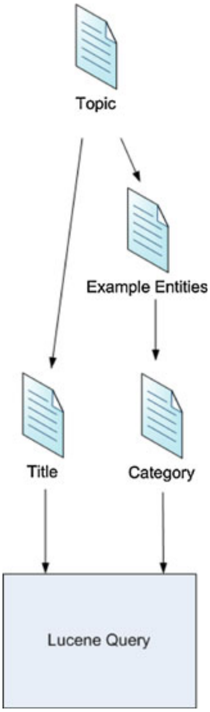
4.1.1 The YAGO ontology

YAGO (Suchanek et al. 2007)[8] is a large and extensible ontology that builds on entities and relations from Wikipedia. Facts in YAGO have been automatically extracted from Wikipedia and unified with semantics from WordNet,[9] achieving an accuracy of around 95%. All objects (e.g., cities, people, even URLs) are represented as entities in the YAGO

[8] Available for download at http://www.mpi-inf.mpg.de/yago-naga/yago/downloads.html.

[9] http://www.wordnet.princeton.edu/.

Fig. 5 Processing pipeline for the list completion system



model. The hierarchy of classes starts with the Wikipedia categories containing a page and relies on WordNet's well-defined taxonomy of homonyms to establish further *subClassOf* relations. We make use of these *subClassOf* relations in YAGO, which provide us with semantic concepts describing Wikipedia entities. For example, knowing from Wikipedia that *Married… with Children* is in the category *Sitcoms*, we reason using YAGO's WordNet knowledge that it is of the type *Situation Comedy*, same as *BBC Television Sitcoms*, *Latino Sitcoms*, *Sitcoms in Canada*, and eight more.

We have implemented two approaches for entity ranking in Wikipedia. Both approaches extend the traditional IR vector space model, enriching it with semantic information. Additionally to textual information from Wikipedia articles we also keep context information (i.e., category information) either extracted from Wikipedia or inferred using YAGO. The examples in the following sections are based on the topic described in Table 1.

4.1.2 Category expansion

While the category information which is present in the topic should contain most of or all the retrievable entities, this is for many topics not the case. Wikipedia is constructed manually by different contributors, so that the category assignments are not always consistent. Many categories are very similar and in some of these cases the difference is very subtle so that similar entities are sometimes placed in different categories by different contributors (e.g., hybrid powered automobiles are, inconsistently, either in the "hybrid vehicles" or the "hybrid cars" category and very seldom they are in both).

In the previous approach the category given in the topic was used to make the query retrieve entities from within that category. The method described here constructs an additional list of categories closely linked to the ones given in the topic description. This extended list of categories is then used instead of the topic categories in query construction. The simplest starting point would be using merely Wikipedia subcategories looking at the Wikipedia categories hierarchy. Apart from this, we use three different types of category expansion, *Subcategories*, *Children* and *Siblings*.

Wikipedia itself has a hierarchical structure of categories. For each category we are presented with a list of *Subcategories*. This list of Subcategories is taken as-is and added to the query. For example, some of the subcategories for the "Actors" category are: "Animal actors", "Child actors", "Actors with dwarfism", "Fictional actors". More in detail, for this approach and the selected topic (see Table 1), the query would have additional *subcategories* as presented in Tables 2 and 3 added to the category search.

The *Children* list of categories is created by starting from the *Subcategories* list and filtering inappropriate ones out. It is more effective not to include all the Wikipedia subcategories in our *Children* list as some of them are not real subcategories, that is, they are not of the same type. As subcategories for a country, it is possible to have categories about presidents, movie stars, or other important persons for that country. This means that

Table 2 Category expansion for topic #109: "National capitals situated on islands"

Categories	Capitals
Subcategories	Capitals Europe, capitals Asia, capitals north America, capitals oceania, english county towns, capitals south america, historical capitals, capitals Africa
Children	Capitals Europe, capitals Asia, capitals north America, capitals oceania, english county towns, capitals south America, historical capitals, capitals Africa
Siblings	Cantonal capitals Switzerland, capitals Africa, capitals Asia, capitals central America, capitals Europe, capitals north America, capitals oceania, capitals south America, capitals Caribbean, former US state capitals, etc.

Children are the result of filters on the subcategories, which may or may not remove terms from the subcategories

Table 3 Category expansion for topic #124: "Novels that won the Booker Prize"

Categories	Novels
Subcategories	Novels by author, **novel sequences**, novels by genre, **novelists**, novels by country, graphic novels, novels by year, novels based on computer and video games, modern library 100 best novels, **warcraft books**, first novels, light novels, autobiographical novels, r.o.d, sequel novels, forgotten realms novels
Children	Novels by author, novels by genre, novels by country, graphic novels, novels by year, novels based on computer and video games, modern library 100 best novels, first novels, light novels, autobiographical novels, r.o.d, sequel novels, forgotten realms novels
Siblings	1902 Novels, 1922 novels, 1947 novels, 1965 novels, 1975 novels, 1994 novels, agatha christie novels, alistair maclean novels, alternate history novels, American novels, Anglo-welsh novels, anne mccaffrey novels, anthony trollope novels, arthur hailey novels, Asian saga novels, Australian novels, autobiographical novels, bret easton ellis novels, british novels, etc.

Children are the result of filters on the subcategories, which may or may not remove terms from the subcategories although we have as a starting category a country we end up having people as subcategories, which is not what we want in the entity retrieval context. The solution to this is selecting only those subcategories having the same class as the initial category. As described in Sect. 4.1, YAGO contains also class information about categories. We make use of this *subClassOf* information to identify suitable categories of the same type. Thus, a Wikipedia subcategory is included in the *Children* list only if the intersection between its ancestor classes and the ancestor classes in YAGO (excluding top classes like *entity*) of the initial category is not empty. The final list of *Children* will therefore contain only subcategories of the same type as the category given in the topic. Figure 6 presents an example of the *Children* list of the category "Sitcoms". For the selected topic (see Table 1), due to the fact that all the *Children* categories have the same type as the topic category, none of them are filtered and the query looks the same as for the *Subcategories* approach (see Table 2). Table 3 presents an example where the subcategories in bold are filtered.

Using YAGO we can also retrieve categories of the same type as the starting category, not restricting just to the Wikipedia subcategories. We first determine the type of the starting category using the *subClassOf* relation in YAGO. Knowing this type we construct a list of all the categories of the same type and add them to the *Siblings* set. *Siblings* are, thus, all the categories of the exact same type as the initial category. Figure 7 shows how, starting from the category "Sitcoms", a list of *Siblings* is created.

Figure 8 depicts the inclusion of *Children* and *Siblings* in the query creation process. Constructing the query is done similarly to the naïve approach setting. The difference relies in the category matching part. In the naïve approach we had only the categories given within the topic while in this case we have the additional three lists of *Subcategories*, *Children* and *Siblings*. For the selected topic (see Table 1) the query would be extended with 23 *Sibling* categories, a part of which is shown in Table 2.

The resulting expanded list of categories is then matched against the *categories* field of the index. These extensions allows to find relevant entities with category information (e.g., "conifers" using the *Subcategory* or *Children* approach) different from the one which is present in the topic (e.g., "trees").

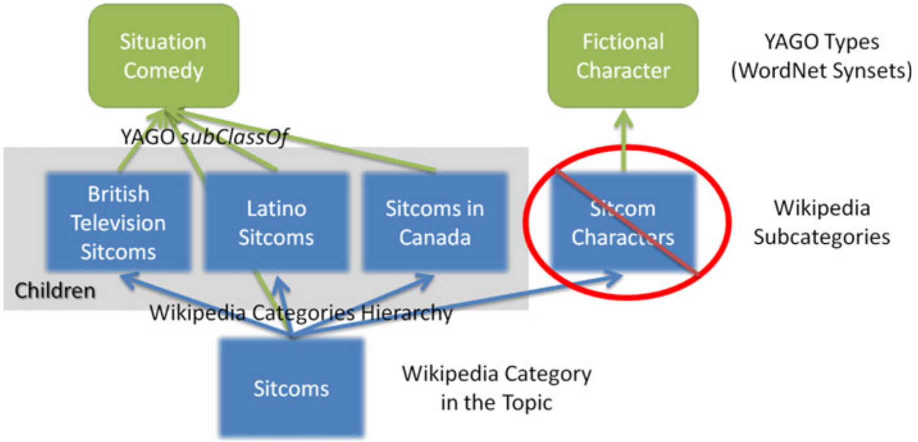


Fig. 6 Example of *Children* identification starting from the "Sitcoms" category

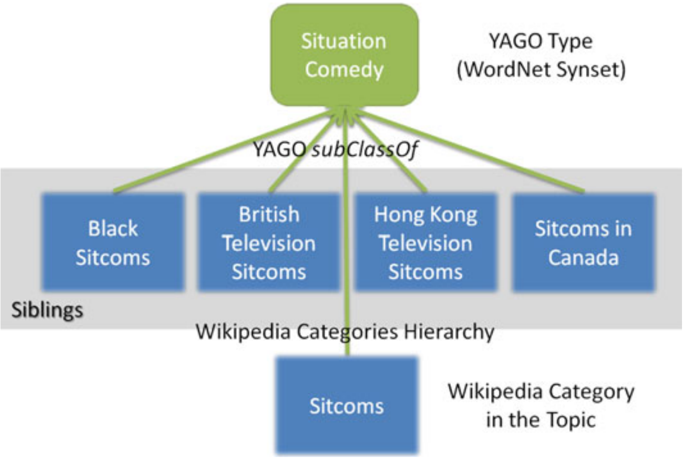
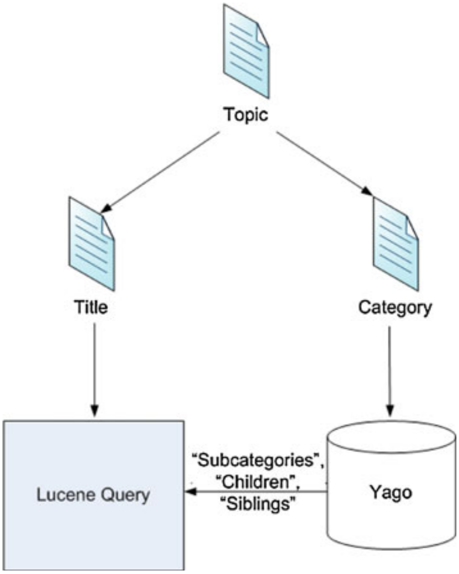

Situation Comedy
YAGO Type (WordNet Synset)
YAGO subClassOf
Black Sitcoms
British Television Sitcoms
Hong Kong Television Sitcoms
Sitcoms in Canada
Siblings
Wikipedia Categories Hierarchy
Sitcoms
Wikipedia Category in the Topic

Fig. 7 Example of *Siblings* identification starting from the "Sitcoms" category

Fig. 8 Query creation using category expansion techniques



4.2 Using Wikipedia links

Wikipedia, just like the Web, is highly interconnected. Search engines make use of link information for traditional IR document ranking. Wikipedia pages, where each page represents an entity, has external links pointing to pages outside the Wikipedia corpus and internal links, which point to other Wikipedia entities. While external links are usually presented in a separate list at the end of the entity description, internal Wikipedia links appear inside the text. While indexing the entity pages, we have kept in the indexed text the names of the linked entities where they appear, and we have also indexed their titles in a separate field called *WikiLinks* to ensure that their importance is not lost in the entity text. In addition to the naïve approach, the contextual part of the query is searched also in the *WikiLinks* index field.

For example, for the query in Table 1 where *London, Tokyo* are relevant results, some of the entities that *London* links to are *Port, Capital City*, whereas *Tokyo* links to *Capital of Japan debate, Izu Islands, Ogasawara Islands* among others. There are many terms present in the list of linked entities, but, as the information in the linked entities field is more condensed than in the text field, linked entities can be a valuable source to improve the ranking of the search results.

4.3 Experimental evaluation

We performed evaluation experiments of our system using the 35 entity topics from the INEX 2008 Entity Ranking Track (see Sect. 3.1). We used the approaches presented in this section and combination of those, with the same notations as used previously, and some additional notations introduced here. Thus, a query is of the form:

$$q = \{(field_i, terms_j)\}$$

where $field_i$ is one of the fields in the Lucene index:

- *text*—the Wikipedia page text;
- *category*—Wiki categories of the pages;
- *outLinks*—outgoing links of the Wiki pages;

and $terms_j$ is a list of terms which should be searched in $field_i$:

- W^X—a list of words given in the topic;
- $Sub(X)$—extract the *subcategories* for the list of words X (e.g., $Sub(W^C)$);
- $Ch(X)$—extract the *children* for X;
- $Sib(X)$—extract the *siblings* for X;

We can combine terms from different approaches: e.g., $q = \{text, W^T \cup W^C\}$, $\{category, W^C\}$ would use the Category from the topic and search this in the Wiki page text together with the Topic Title. Additionally the Topic Category is searched in the Wikipedia categories.

We evaluated our approaches against the *naïve approach* presented in Sect. 3.3 which has an xInfAP value of 0.2350 and a Precision for the first 10 results (P@10) of 0.306. Table 4 shows the first 10 results for topic #109 (“National capitals situated on islands”) using the *naïve approach*. We also show whether the result was assessed and if so whether it was considered relevant. As can be seen, not all relevant entities were assessed as relevant (e.g., capital of Solomon Islands is Honiara).

We performed the experiments on the evaluation dataset and we compared the algorithms which use category information and link information. The results (presented in Table 5) show that using extra category information more than just the one present in the topic does not improve the effectiveness, as the additional categories introduced contain too much noise on average. From the three category expansion techniques, the best performing approach, in terms of xInfAP, is obtained using the Subcategories. What we observed is that the use of Siblings gave even worse performance as the Siblings categories were greater in number than the Subcategories. We also included an approach where no

Table 4 Top 10 results using the naïve approach for topic #109 ("National capitals situated on islands") together with the containing Wikipedia category of each result

Rank	Entity	Most relevant category	Relevance
1	County town	Capitals	0
2	Kinston, Norfolk Island	Capitals in oceania	0
3	Palikir	Capitals in oceania	0
4	Washington Capitals	Washington capitals	0
5	Port Vila	Capitals in oceania	1
6	Belmopan	Capitals in north America, cities in belize	NA
7	Victoria, Seychelles	Capitals in Africa	1
8	Honiara	Capitals in oceania	0
9	Capital	Capitals	0
10	Avarua	Capitals in oceania	1

Relevance is 0 = assessed not relevant; 1 = assessed relevant; NA = not assessed

Table 5 Average precision and precision over the first 10 results for *Categories Based Search* in the XER task

Nr	Method	Query; $q = \ldots$	xInfAP	P@10
1	Baseline	$\{text, W^T\}, \{category, W^C\}$	0.2350	0.3057
2	No category	$\{text, W^T\}$	0.1125*	0.1429*
3	Title as category	$\{text, W^T\}, \{category, W^C \cup W^T\}$	**0.2641***	0.3286
4	Category as title	$\{text, W^T \cup W^C\}, \{category, W^C\}$	0.2190*	0.2571
5	Subcategories	$\{text, W^T\}, \{category, W^C \cup Sub(W^C)\}$	0.1618*	0.2085*
6	Children	$\{text, W^T\}, \{category, W^C \cup Ch(W^C)\}$	0.1616*	0.2057*
7	Siblings	$\{text, W^T\}, \{category, W^C \cup Sib(W^C)\}$	0.1111*	0.1228*
8	Wiki links	$\{text, W^T\}, \{category, W^C\}, \{outLinks, W^T\}$	0.2561*	**0.3399***

The results marked with * are statistically significant (two-tailed t-test, $p < 0.05$) as compared to the baseline

Results in bold have highest scores

category information is searched. Although overall results are worse than the baseline, results improved for topics where the categories are assigned inconsistently in Wikipedia (see Sect. 6 for an analysis).

Another observation we made is that the YAGO ontology is up-to-date and does not match all of the categories present in the XML Wikipedia dataset, which is a crawl of 2005. Thus the evaluation assessments might not consider relevant information which is present today in YAGO.

We also used the internal link structure within Wikipedia in order to improve the results of the search for entities. From the results presented in Table 5 we can see that the simple title search in the outgoing links of a page improves the effectiveness over the baseline by 9% in terms of xInfAP.

For evaluating the impact on the users, we can look at the value of the P@10 which gives an intuition on how many relevant entities the system retrieves at the top (i.e., the part of the results that the user would care most about) while xInfAP evaluates the overall ranking generated by the system.

Table 6 Average precision and precision over the first 10 results for *Categories Based Search* in the LC task

Nr	Method	Query; $q = \ldots$	xInfAP	P@10
1	Baseline	$\{text, W^T\}, \{category, W^{LC}\}$	0.2885	0.3399
2	No category	$\{text, W^T\}$	0.0998*	0.1200*
3	Title as category	$\{text, W^T\}, \{category, W^{LC} \cup W^T\}$	0.2836	0.3342
4	Category as title	$\{text, W^T \cup W^{LC}\}, \{category, W^C\}$	0.2841	0.3428
5	Subcategories	$\{text, W^T\}, \{category, W^{LC} \cup Sub(W^{LC})\}$	0.2466	0.2857
6	Children	$\{text, W^T\}, \{category, W^{LC} \cup Ch(W^{LC})\}$	0.2509	0.2885
7	Siblings	$\{text, W^T\}, \{category, W^{LC} \cup Sib(W^{LC})\}$	0.1108*	0.1171*
8	Wiki links	$\{text, W^T\}, \{category, W^{LC}\}, \{outLinks, W^T\}$	**0.3006**	**0.3685***

The results marked with * are statistically significant (two-tailed *t*-test, $p < 0.05$) as compared to the baseline

Results in bold have highest scores

For completeness, we performed the experiments for the LC task, where the starting categories are extracted from the topic example entities. The results, presented in Table 6, are consistent with those of the XER task.

5 NLP based techniques for entity retrieval

In this section we present our algorithms for improving ER in Wikipedia using Information Extraction (IE) and Natural Language Processing (NLP) techniques (Demartini et al. 2008).

For comparison reasons, we also search the textual part of the query in the *outLinks* index field as presented in Sect. 4. This approach can easily be combined with others to improve performance (e.g., searching the Topic Title in the *text* field AND in the *outLinks* field).

5.1 Using lexical compounds

Anick and Tipirneni (1999) defined the *lexical dispersion hypothesis*, according to which an expression's lexical dispersion (i.e., the number of different compounds it appears in within a document or group of documents) can be used to automatically identify key concepts in the input document set. Although several possible compound expressions are available, it has been shown that simple approaches based on noun analysis are almost as good as highly complex part-of-speech pattern identification algorithms (Allan and Raghavan 2002). Verbs, for example, are not very helpful since they are typically too general and used in a variety of different contexts. Lexical Compounds have been already used in different settings for refining web search queries (Chirita et al. 2007). We thus extract from simple text all the Lexical Compounds of the following form: $\{adjective?\ noun+\}$. All such compounds could be easily generated for Title in Topics using WordNet. Moreover, once identified, they can be further sorted depending on their dispersion within each topic. We then use Lexical Compounds as search terms in the query, as they present the essential information in a more concise manner. We consider two approaches to using Lexical Compounds in constructing the query. The first uses only

Lexical Compounds for constructing the textual part of the query, to search over all the text index field. In the second approach we use the text from the topic title along with the extracted Lexical Compounds to search over the *textstem* field. For example, for the Title of the topic in Table 1, *National capitals situated on islands*, our algorithm extracted three Lexical Compounds: *national capitals*, *islands* and *national*.

5.2 Synonyms and related words

Wikipedia, just as the general Web, presents its information in natural language. There is no formal representation and only limited structured information. After describing how to use the structured information, like category information or link structures, we examine different approaches exploiting *natural language properties*.

The first approach accommodates the fact that there are various ways of conveying the same meaning within natural language sentences or even words. This observation lead us to the conclusion that only using the present keywords in the Title, Description, or Category fields is not enough. Therefore, starting from previous research on query expansion through WordNet synonyms (Voorhees 1993, 1994; Hsu et al. 2006; Bhogal 2007 etc.) we extended the query using *related words* and *synonyms* of the extracted keywords.

To add the correct synonyms and related words to the query we need to identify the nouns of a query. For this we use part-of-speech tagging from LingPipe (Alias-i2008)—a suite of java libraries for NLP. The part-of-speech tagger was trained on the manually labelled Brown corpus, a collection of various types of text documents, to obtain statistical models to perform part-of-speech tagging.

The synonyms and related words were automatically generated using the WordNet semantic lexicon (Fellbaum 1998). WordNet can be seen as a dictionary that groups English words into sets of synonyms and stores the various semantic relations between these synonym sets (*synsets*). As there are several synsets available for each term in WordNet, we first perform Word Sense Disambiguation, as done in Semeraro et al. (2007), to choose the correct meaning for the nouns in the query. Then we extend the query with additional information about each noun: (1) add all synonyms from the previously identified synset; (2) add all words that have a relationship (except for antonyms) to the identified synset. The additional words are then used to enrich the query to improve the recall of our system:

$$w_i \in W^T \cup Synonyms(W^T) \;\text{ or }\; w_i \in W^T \cup RelatedWords(W^T)$$

5.3 Core characteristics

To make the query more precise, we examined the results for removing parts of the query. On the one hand we *removed duplicate information* in the title by finding synonym nouns occurring in the category field. This was achieved using WordNet as described in 2. Since we try to find entities and not categories, the idea is to remove category keywords from the query. Making use of synonym information makes this approach more robust and helps to extract core characteristics from the user query. On the other hand we used LingPipe's part-of-speech Tagger to identify verbs, nouns, adjectives, etc. and removed all except *nouns* and *adjectives*. Observations showed that nouns and adjectives are especially helpful to describe entities, whereas verbs mostly introduce noise to the results due to their generality. The formal notation for this approach is:

$$w_i \in W^T_{Adj} \cup (W^T_{Nouns} \setminus (W^C \cup Synonyms(W^C))$$

5.4 Named entity recognition

Another well known concept in IE is *Named Entity Recognition*. The knowledge about named entities in the query can be a valuable hint to identify what kind of entity is expected in the answer. We use Named Entity (NE) Recognition provided by LingPipe. Finding named entities can be done using dictionary matching, regular expressions, or statistical approaches. We used a machine learning approach with a model gained from supervised training on a large news article corpus. We identified different named entities like *organizations*, *locations*, and *persons*. The found named entities were then used to perform a keyword search using the following terms:

$$w_i \in W^T \cap \{NamedEntities\}$$

Table 7 shows an example of the different approaches for topic #109 *National capitals situated on islands*.

5.5 Experimental evaluation

Similarly to the previous evaluation methodology, presented in Sect. 4.3, we used the Wikipedia collection provided by INEX. We used the approaches presented in this section and combination of those, with the same notations as used previously, and some additional notations introduced here. Thus, a query is of the form:

$$q = \{(field_i, terms_j)\}$$

where $field_i$ is one of the fields in the Lucene index:

Table 7 Topic #109 after applying the different strategies

Title	National capitals situated on islands
Category	Capitals
Synonyms	Capitals islands on National “working capital” situated
Related words	**Synonyms** plus additional concepts related mainly to capitals capitals Bahrein “Galveston Island” “Hawaii Island” “Molokai Island” Kriti “Faroe Islands” Zanzibar Haiti Anglesey ‘Vancouver Island” Nihau Corse Ceylon Kahoolawe Moluccas “South Island” Papua Hibernia Hispaniola “seed money” Saba “Aegadean Islands” “St. Kitts” “Saint Lucia” “Visayan Islands” “Puerto Rico” Sulawesi Iceland “New Zealand” Curacao Guadeloupe Barbados “Spice Islands” “St. Martin” “Netherlands Antilles” Sicilia “British Isles” Azores “Aran Islands” Tobago “quick assets” Montserrat Formosa Hondo “Falkland Islands” “Martha’s Vineyard”Maui situated GU isle Crete Bisayas “risk capital” Honshu “Republic of China” Anglesea “Wake Island” Taiwan “Kodiak Island” Mindoro Maldives “Viti Levu” “Canary Islands” Fijis Krakatao “St. Eustatius” “solid ground” Cyprus “Maui Island” Krakatau Vieques Principe Hokkaido Bali Bougainville “Baffin Island” Borneo Bonaire “Oahu Island” Staffa “Isle of Man” Kodiak Kalimantan assets “Catalina Island” “Kahoolawe Island” Corsica Okinawa Saipan Ithaki
Core characteristics	National
Named entities	National islands

- *text*—the Wikipedia page text;
- *title*—the Wikipedia page title;
- *category*—Wiki categories of the pages;
- *outLinks*—outgoing links of the Wiki pages;

and $terms_j$ is a list of terms which should be searched in $field_i$:

- W^X—a list of words given in the topic;
- $LexComp(X)$—extract the *Lexical Compounds* from X;
- $SY(X)$—apply the *synonyms* approach on the list of words X (e.g., $SY(W^T)$);
- $RW(X)$—apply the *related words* approach on X;
- $NE(X)$—extract only the *named entities* from X;
- $CC(X)$—apply the *core characteristics* approach on X;
- $X \cup Y$—union of all terms in X and Y;
- $+X$—all terms in X have to be present in the searched field (conjunction);
- $-X$—all terms in X must *not* be present in the searched field (negation);

Table 8 presents the Average Precision (xInfAP) and Precision for the first ten retrieved results (P@10) of our approaches. Additional to the query presented for each approach, the Category given with the topic was also searched in the *category* field of the index. The

Table 8 Average precision and precision for the first 10 results for NLP based techniques for the XER task

Nr	Query; $q = \{category, W^C\} \cup \ldots$	xInfAP	P@10
1	$\{text, W^T\}$	0.2350	0.3057
9	$\{text, W^T\}, \{outLinks, W^T\}$	0.2556*	0.3371*
10	$\{text, W^T\}, \{outLinks, CC(W^T)\}$	0.2511	0.3114
11	$\{text, W^T\}, \{outLinks, NE(W^T)\}$	0.2504*	0.3171
12	$\{LexComp(W^T)\}$	0.2284	0.2971
13	$\{text, W^T \cup LexComp(W^T)\}$	0.2506	0.3257
14	$\{text, W^T \cup LexComp(W^T)\}, \{outLinks, W^T \cup LexComp(W^T)\}$	0.2616	0.3457
15	$\{text, W^T \cup SY(W^T)\}$	0.2439*	0.3257
16	$\{text, W^T \cup RW(W^T)\}$	0.2398	0.3199
17	$\{text, W^T \cup CC(W^T)\}$	0.2509*	0.3257
18	$\{text, W^T \cup NE(W^T)\}$	0.2530*	0.3257
19	$\{text, W^T \cup SY(W^T) \cup RW(W^T) \cup CC(W^T) \cup NE(W^T)\}$	0.2705*	0.3571*
20	$\{text, W^T \cup SY(W^T) \cup RW(W^T) \cup CC(W^T) \cup NE(W^T)\}, \{outLinks, CC(W^T)\}$	0.2682*	0.3599*
21	$\{text, W^T \cup SY(W^T) \cup RW(W^T) \cup CC(W^T) \cup NE(W^T)\}, \{category, W^T\}$	**0.2909***	**0.3971***
22	$\{text, +W^T \cup SY(W^T) \cup RW(W^T) \cup CC(W^T) \cup NE(W^T)\}$	0.0813*	0.1124*
23	$\{text, W^T \cup SY(W^T) \cup RW(W^T) \cup + CC(W^T) \cup NE(W^T)\}$	0.2627	0.3857
24	$\{text, W^T \cup SY(W^T) \cup RW(W^T) \cup CC(W^T) \cup NE(W^T)\}, \{outLinks, CC(W^T)\}, \{title, -W^T\}$	0.2748*	0.3657*
25	$\{text, W^T \cup SY(W^T) \cup RW(W^T) \cup CC(W^T) \cup NE(W^T)\}, \{outLinks, CC(W^T)\}, \{title, -W^C\}$	0.2534	0.3314

The results marked with * are statistically significant (two-tailed t-test, $p < 0.05$) as compared to the baseline (#1)

Results in bold have highest scores

baseline used is approach #1 with a Average Precision and P@10 values of 0.2350 and 0.3057.

We evaluated our algorithms both independently and as combinations of several approaches. All our approaches improved in terms of both Average Precision and P@10 over the baseline, with the combination of all approaches showing the highest improvement. When compared to the official submissions at INEX-XER 2008[10] our best run with a xInfAP score of 0.29 would place us third participating group as the second best score used also example entities and, therefore, can not be compared. The best performing system at INEX-XER 2008 (Vercoustre 2009) obtained a score of 0.341 by learning system parameters based on the topic difficulty which is an orthogonal approach to the ones presented in this paper.

For completeness, in Table 9 we present the results for the LC task. The results are again consistent with the XER task. As the LC task was not the focus of our research, we tried only a simple approach for learning the categories and our best result ranks in the middle of the LC runs submitted at INEX-XER 2008. In the following we discuss how different approaches designed for the XER task performed.

5.5.1 Outgoing links

Approaches #9, #10, #11 from Table 8 show the results for searching with the terms from the topic Title, with the Core Characteristics, and with the Named Entities approaches in the outgoing links text of Wikipedia pages, respectively. The simple (#9) approach shows 10% improvement in P@10 over the baseline. This proves extracting concept names (done as outgoing links in Wikipedia) from entity descriptions to be a valuable additional information for raising early precision values.

5.5.2 Lexical compounds

In order to evaluate the approaches based on syntactic information we extracted the Lexical Compounds from the Topic Title and we performed several comparisons. The results are presented as #12, #13, #14 in Table 8. The simple extraction of Lexical Compounds from the topic does not show improvements but it is possible to see that the combined usage of Lexical Compounds and the Topic Title performs better than the baseline. Combining the most promising approaches (i.e., searching in the outgoing links and using Lexical Compounds together with topic terms) improves the results even more.

5.5.3 Synonyms and related words

Adding only synonyms of nouns (#15) results in better performance than adding all related words of the nouns (#16). This is due to the vast amount of noise added by RW. Also SY adds some noise as although Word Sense Disambiguation was performed prior to adding the synonyms, still some synonyms are misleading and might need a further filtering step.

5.5.4 Core characteristics

Approach #17, when used for searching in the whole page text shows the same level of improvement as RW. But when combining it with other methods it improves results

[10] http://www.l3s.de/~demartini/XER08/.

Table 9 Average precision and precision for the first 10 results for NLP based techniques for the LC task

Nr	Query; $q = \{category, W^{LC}\} \cup \ldots$	xInfAP	P@10
1	$\{text, W^T\}$	0.2885	0.3399
9	$\{text, W^T\}, \{outLinks, W^T\}$	0.3006	0.3685*
10	$\{text, W^T\}, \{outLinks, CC(W^T)\}$	0.2995	0.3657*
11	$\{text, W^T\}, \{outLinks, NE(W^T)\}$	0.2919*	0.3571*
12	$\{LexComp(W^T)\}$	0.3011	0.3628*
13	$\{text, W^T \cup LexComp(W^T)\}$	0.3054	0.3571
14	$\{text, W^T \cup LexComp(W^T)\}, \{outLinks, W^T \cup LexComp(W^T)\}$	0.2872	0.3914*
15	$\{text, W^T \cup SY(W^T)\}$	0.3020*	0.3486*
16	$\{text, W^T \cup RW(W^T)\}$	0.2969	0.3342
17	$\{text, W^T \cup CC(W^T)\}$	0.3012*	0.3599*
18	$\{text, W^T \cup NE(W^T)\}$	0.2979	0.3543
19	$\{text, W^T \cup SY(W^T) \cup RW(W^T) \cup CC(W^T) \cup NE(W^T)\}$	0.3187*	0.3771*
20	$\{text, W^T \cup SY(W^T) \cup RW(W^T) \cup CC(W^T) \cup NE(W^T)\}, \{outLinks, CC(W^T)\}$	0.3116	0.3743*
21	$\{text, W^T \cup SY(W^T) \cup RW(W^T) \cup CC(W^T) \cup NE(W^T)\}, \{category, W^T\}$	**0.3237**	0.3828
22	$\{text, +W^T \cup SY(W^T) \cup RW(W^T) \cup CC(W^T) \cup NE(W^T)\}$	0.0914*	0.1286*
23	$\{text, W^T \cup SY(W^T) \cup RW(W^T) \cup + CC(W^T) \cup NE(W^T)\}$	0.3221	**0.3914**
24	$\{text, W^T \cup SY(W^T) \cup RW(W^T) \cup CC(W^T) \cup NE(W^T)\}, \{outLinks, CC(W^T)\}, \{title, -W^T\}$	0.3093	0.3686*
25	$\{text, W^T \cup SY(W^T) \cup RW(W^T) \cup CC(W^T) \cup NE(W^T)\}, \{outLinks, CC(W^T)\}, \{title, -W^{LC}\}$	0.2885	0.3486

The results marked with * are statistically significant (two-tailed t-test, $p < 0.05$) as compared to the baseline (#1)

Results in bold have highest scores

significantly. The average numbers for this case are misleading since, e.g., approach #23 accounts for a couple of top results on a per topic evaluation. This shows extracting the key concepts from both the page text and the query text as being useful for improving early precision.

5.5.5 Named entity recognition

Similar to CC, NE (#18) shows statistically significant improvements of 8% for Average Precision. We see that searching with more weight (i.e., duplicating NE words, as they already appear in the Topic Title) the named entities helps improving the ranking.

5.5.6 Combining the approaches

All approaches improve but each ranks entities differently. This leaves room for improvement by combining the single approaches. We performed several combinations and present only the best performing ones. When searching in the page text, we found that including all methods in the query (#19) improves Average Precision by 15% and P@10 by 17%. Adding category (#21) improves even more and reaches 0.291 Average Precision and 0.397 P@10; both with a statistically significant difference with the baseline. This is an improvement of 24 and 30%, respectively.

5.5.7 Efficiency considerations

We have implemented all the presented approaches in the Java programming language. Our test system is an Intel(R) Core(TM)2 CPU (2GHz) and 2 Gbyte RAM, running with *Ubuntu Karmic OS (Kernel 2.6.31-16-generic).* As Java compiler we use OpenJDK Runtime Environment (IcedTea6 1.6.1). The Lucene index with all the Wikipedia records has a size of 6.0 GByte. Each run (for 35 topics) took on average 0.987 min (59.271 s), when the system ran single threaded using only one CPU and 1Gbyte RAM.

6 Results and discussion

The size of the corpus available is 35 topics. This is not enough to cover all facets of Entity Retrieval or give a complete overview of all different types of entity related queries. It is, however, possible to identify certain patterns which influence the performance of different algorithms on the used test collection and identify different types of queries.

6.1 Wikipedia categories vs. topic categories

Category information can be very useful to identify the information needed by the user. Unfortunately, the given category from the user and the existing categories in Wikipedia do not always match as expected. Approaches to solve this problem have been proposed in the first two editions of INEX-XER. For example, in Tsikrika et al. (2008) the authors propose to walk the category hierarchy graph up to three levels starting from the given topic category in order to find other possible Wikipedia categories that best match the information need. In the following we analyse INEX-XER 2008 topics with the goal of understanding which topic categories can be used directly and which need to be automatically refined before the retrieval step.

6.1.1 Correctly assigned categories

The analysis showed that for different types of queries particular approaches perform well while others perform worse. In general we identified on the one hand a set of queries which yielded good results for most of the systems participating at INEX. These “easy” queries have the property that at least one of their assigned topic categories are rather specific categories in Wikipedia as opposed to general categories.

By comparing the scores of the text only approach and the baseline approach (text and category search) we discovered that there are 21 topics for which the given categories help when searching. For eight of these topics, the improvement when using the category information is over 40% xInfAP. When analysing the categories of these topics, we noticed that the given categories are topic specific and also a high ratio of the pages assigned to them are relevant for the query. By specific categories we mean categories that have few pages assigned. For example, for the topic #140, the category *airports in germany* has only 30 pages assigned to it, out of which 28 have been assessed as relevant. Also, for topic #114, category *formula one grands prix* has 27 pages assigned with 15 being relevant.

On the other hand, when the topic categories were too general with respect to the query (i.e., with many pages assigned to them and few of these pages being relevant to the topic), we observed that searching with the text only performed better than using also the category information. For example, for topic #104, the category *harry potter characters* (with 10 relevant out 114 assigned pages), proved to be too general when actually searching for characters that were on the Gryffindor Quidditch team.

6.1.2 Misleading and inconsistent category assignments

In the topic set we identified six topics that have low performance (xInfAP smaller than 25%) for all the systems participating at INEX. From these "difficult" topics we noticed that three had categories with no or few pages assigned to them. For example, for topics #106, #125 and #133, the categories *peerage of england* and *country* are empty and whereas *countries* has only two pages assigned to it. For two of these "difficult" topics the categories have been assigned wrong at topic creation time. Thus, for topic #147, the category *eponyms* denotes people, when the topic is about chemical elements. Also, for topic #118, about car models, the specified categories are about car manufacturers.

Another observation is that the topic categories are usually either denoting the type of the desired entities or some property of the relevant entities, or a combination of the two. What happens usually is that the property type categories are too general for the topics, thus hindering the ER performance, as they lead to the retrieval of many different types of entities. For example, category *harry potter* when searching for characters in topic #104 contains no relevant pages in our test corpus. Also, categories such as *hundred years war* when looking for people in topic #106, and *geography of switzerland* for cantons in topic #119, are not particularly helpful on their own.

Interestingly, for the topic #127 ("german female politicians"), where both categories (*politicians*, *women*) represent indeed the types of the desired entities, all the systems had poor performance. This happened because both of the categories had no relevant entities in the Wikipedia corpus used in the experiments. They are all similar to the relevant Wikipedia category, e.g., "bond girls" for topic #128. Also two third of the easy queries is about persons. On the other hand we have queries which none of the systems could answer satisfactorily. These queries share a broad, respectively general category information. Best performing group on difficult topics is cirquid, see Rode (2009): expanding category information walking three steps in the category hierarchical graph, using thus also the categories that are ancestors to the topic categories. This improves then results for topics which had initial categories unrepresented in the INEX Wikipedia corpus (i.e., the categories were empty or had few entities).

6.2 Query terms

Several queries contain certain terms which can change the query meaning almost completely, but are not perceived as such by a classical IR system. E.g., by requesting "*living nordic* classical composers" instead of just "classical composers" puts a very high restriction on the query but the means by which an IR system can identify important terms (e.g., Inverse Document Frequency—IDF) will not rate much higher documents containing "nordic composers" instead of "classical composers". The same applies also for "fifa world cup national team *winners* since 1974" where "winners", from the point of view of the system, is just one term out of eight which are similarly important.

6.3 Time span queries

Certain queries restrict the result set based on time spans, e.g., "French car models in the 1960s". These restrictions require special attention on dates, in query analysis as well as in analysis of potential result pages. To improve precision, systems need to be more time sensitive. The results show that queries with time constraints are difficult for all systems.

6.4 Winning approaches and topic types

We have grouped the topics into four sets based on the query methods that had the maximum performance, see Table 10 for an overview of the best approaches per topic.

6.4.1 Method 2: ignoring category information

When not using the topic categories and searching only with the topic titles we had maximum performance for six of the 35 topics. These topics were had categories that were either too general (topics #104, #112) or wrongly assigned (topics #118,#147). For example, category *guitarist* has 501 entities assigned to it and out of this only 22 were assessed as relevant to the topic #112.

6.4.2 Methods 3 and 21: using the title as category

For the methods where we additionally searched with the topic titles in the *category* field, we had improvement on eight topics. This usually happened when the topic title had additional content words that have been used as in category names in the Wikipedia corpus. For example, for the topic #136 with category *baseball players*, the additional words in the title would be *Major League Baseball*. There 10 more categories related to this in the corpus, three of these are related to players. Also, for some of the topics, the title can contain synonyms to words in the category, e.g., for topic #141, we have *Catalunya* in the title and *catalan* in the category.

6.4.3 Methods 22 and 23: requiring all terms to be matched

When using the NLP techniques that we have mentioned in the previous section we introduce also noise in the queries. Thus, when we make restrictions such as “the result must contain all words from the topic title” or “the result must contain all words from the topic title core characteristics”, we give high importance to keywords that might otherwise get lost in the NLP query. This approaches work mostly on short titles, or on topics where the core characteristics are meaningful.

6.4.4 Methods 24 and 25: negating terms in Wiki titles

One of the main issues that we noticed in our experimental results was that we retrieved many non-entity pages. We tried to filter these out by restricting results as to not to contain keywords from the title or category in the result name. This improved the performance on four topics, by filtering out additional related but yet not relevant pages. For example, for topic #130 with the title “star trek captains” and having as categories three of the Star Trek movie titles, there are at least 159 pages that where excluded. All this excluded entities contained the keywords “star trek” in their title as means of disambiguation, that they appeared in the “Star Trek” environment. Characters from the movie (e.g., *Jean-Luc Picard*) do not contain the movie title in their names.

Table 10 Effectiveness values (xInfAP) on each INEX-XER 2008 topic

ID	Title [categories]	xInfAP	Method
104	Harry Potter Quidditch Gryffindor character [harry potter, harry potter characters]	0.5397	2
106	Noble english person from the Hundred Years' War [peerage of England, 100 years war]	0.1952	20
108	State capitals of the United States of America [U.S. state capitals, capitals, capital cities]	0.6789	23
109	National capitals situated on islands [capitals]	0.2567	23
110	Nobel prize in Literature winners who were also poets [nobel prize in literature winners]	0.5936	21
112	Guitarists with mass-produced signature guitar models [guitarists]	0.1241	2
113	Formula 1 drivers that won the Monaco Grand Prix [racecar drivers, formula one drivers]	0.1879	23
114	Formula one races in Europe [formula one grands prix]	0.5058	16
115	Formula One World Constructors' Champions [formula one constructors]	0.4014	23
116	Italian nobel prize winners [nobel laureates]	0.4663	23
117	Musicians who appeared in the Blues Brothers movies [musicians]	0.0838	23
118	French car models in 1960s [automobile manufacturers, French automobile manufacturers]	0.0341	2
119	Swiss cantons where they speak German [geography of Switzerland, cantons of Switzerland]	0.8853	24
121	US presidents since 1960 [presidents of the United States, U.S. democratic party presidential nominees, U.S. republican party presidential nominees]	0.3224	22
122	Movies with eight or more Academy Awards [best picture oscar, british films, American films]	0.1585	2
123	FIFA world cup national team winners since 1974 [football in Brazil, European national football teams, football in Argentina]	0.094	2
124	Novels that won the Booker Prize [novels]	0.4646	23
125	Countries which have won the FIFA world cup [countries]	0.1111	23
126	Toy train manufacturers that are still in business [toy train manufacturers]	0.2573	21
127	German female politicians [politicians, women]	0.1088	3
128	Bond girls [film actors, bond girls]	0.7294	22
129	Science fiction book written in the 1980 [science fiction novels, science fiction books]	0.3969	3
130	Star Trek Captains [star trek: the next generation characters, star trek: voyager characters, star trek: deep space nine characters]	0.1705	25
132	Living nordic classical composers [twenty first century classical composers, Finnish composers, living classical composers]	0.0523	4
133	EU countries [country]	0.0663	23

Table 10 continued

ID	Title [categories]	xInfAP	Method
134	Record-breaking sprinters in male 100-m sprints [sprinters]	0.1339	19
135	Professional baseball team in Japan [Japanese baseball teams]	0.7453	18
136	Japanese players in Major League Baseball [baseball players]	0.3107	3
138	National Parks East Coast Canada US [national parks, national parks of the United States, national parks of Canada]	0.2286	3
139	Films directed by Akira Kurosawa [Japanese films]	0.9198	24
140	Airports in Germany [airports in Germany]	0.9912	10
141	Universities in Catalunya [catalan universities]	0.7058	21
143	Hanseatic league in Germany in the Netherlands Circle [cities, cities in Germany]	0.215	9
144	Chess world champions [chess grandmasters, world chess champions]	0.7167	3
147	Chemical elements that are named after people [eponyms]	0.0469	2

We report our best results along with the best performing approach for each topic. See Tables 5 and 8 for the methods' descriptions

6.5 Discussion

In this section we have presented an analysis of system performance on the different topics available in the test collection. We have shown how it is important for systems to find good categories also by walking the Wikipedia category graph.

We have seen how easy-to-answer topics have specific categories with few pages assigned to them and that categories contain the desired type of entity (e.g., persons). Difficult topics have categories which are empty or too general, that is, they contain different types of entities in it. It is also important for systems to identify key terms in the query (e.g., "living", "winners") and to process queries containing time spans (e.g., "since 1974").

We have also seen how for some topics it is necessary to ignore the category information provided and for others to use the query for searching the *category* field as the query contains information about the desired entity type. For short queries we have seen that it helps performing phrase queries.

While we only analyzed different types of topics used in entity retrieval, creating an adaptive system depending on user input is the logical next step to pursue. Similar to Vercoustre et al. (2009), the system should employ different algorithms and ranking criteria according to the types of topics identified previously.

7 Related work

Finding entities on the Web is a recent topic in the IR field. The first proposed approaches (Bast et al. 2007; Cheng and Chang 2007; Cheng et al. 2007) mainly focus on scaling efficiently on Web dimension datasets but not on the effectiveness of search, as addressed in this paper.

A formal model for entities has been presented in Palpanas et al. (2008). This entity representation is, similarly to our proposal, based on (<*attribute*>, <*value*>) pairs and on a "Category of reference" that describes the entity type which can be taken from an ontology. In our paper we propose a model for the entire ER process where the entity representation is just a sub-part. A framework for modelling the IR process has been presented in Rölleke et al. (2006) where the authors present a matrix-based framework for modelling possible search tasks. The model we propose is focused on ER; it is less formal but more intuitive.

Approaches for finding entities have also been developed in the Wikipedia context. Previous approaches to rank entities in Wikipedia exploited the link structure between Wikipedia pages (Pehcevski et al. 2008) or its category structure using graph based algorithms (Tsikrika et al. 2008). Compared to these approaches, we start first designing a model for ER making the development of algorithms possible also in domains different from Wikipedia and we exploit semantic and NLP techniques to improve effectiveness. Our next step will be to apply the algorithms, evaluated on the Wikipedia corpus, on the entire Web, as done in Bast et al. (2007), Cheng and Chang (2007), Cheng et al. 2007), aiming to find the best compromise between efficiency and effectiveness of search.

With respect to previous approaches we based our algorithms on a structured representation of entities at indexing level—we used a structured index built using NLP techniques. For this reason, relevant to our work are projects aiming at extracting and annotating entities and structure in Wikipedia. For example, versions of Wikipedia annotated with state of the art NLP tools are available (Schenkel 2007; Jordi Atserias 2008).

Another relevant work is Zaragoza et al. (2007) which also aims at retrieving entities in Wikipedia but without the assumption that an entity is represented by a Wikipedia page as done in INEX-XER. They rather annotate and retrieve any passage of a Wikipedia article that could represent an entity. Our structured index allows such kind of retrieval as well. A foundation for an effective ER can also be the automatic identification of instances and classes in the Wikipedia category hierarchy (Zirn et al. 2008). Knowing which categories describe instances can help the ER system in finding entities relevant to the query because not all the articles in Wikipedia are entity descriptions.

An important related area of research is entity identity on the Web. It is crucial for the ER task being able to uniquely and globally identify entities on the Web so that the search engine can return a list of identifiers to the user who can afterwords navigate in the entity descriptions. A strong discussion already started in the Web research community (Bouquet et al. 2007, 2008) and solutions for entity identity resolution on the Web have been proposed (Bouquet et al. 2008). Our solution for finding entities relies on these infrastructures able to globally identify entities on the Web.

With respect to our final analysis of easy and difficult topics, a related area is that of query difficulty prediction (Carmel et al. 2005). In particular, in Vercoustre et al. (2009) they study how to automatically predict the difficulty of an ER query in the Wikipedia context. They also study how to adapt their system variables accordingly in order to improve effectiveness. Our findings about what characterizes a difficult or easy topic are consistent with the features they use for classifying topics.

For example, they use the number of articles attached to categories, the number of categories attached to the entities, query length, etc. Compared to this work we perform a more detailed analysis of which properties make a query difficult or not for systems to answer. On the other hand, we did not make our system adaptive to different topics even though we have shown how different techniques among the proposed ones work better for different topics.

8 Conclusions and further work

In this paper we presented a general model for ranking entities and we showed how the model can be applied to different real world scenarios. We described in detail a possible instantiation of the model and a set of algorithms designed for the Wikipedia dataset. We make use of the Wikipedia structure—page links and categories—and employ an accurate ontology to remove possible noise in Wikipedia category assignments. The results show that, in the used test collection, category assignments can be both very helpful for retrieval as well as misleading depending on the query syntax. We also employ several NLP techniques to transform the query and to fill the gaps between the query and the Wikipedia language models. We extract essential information (lexical expressions, key concepts, named entities) from the query, as well as expand the terms (by means of synonyms or related words) to find entities by specific spelling variants of their attributes. By combining several techniques we can achieve a relatively high effectiveness of the ER system; still, further improvement is possible by selectively applying the methods for different queries. The experimental evaluation of the ER algorithms has shown that by combining our approaches we achieve an average improvement of 24% in terms of xInfAP and of 30% in terms of P@10 on the XER task of the INEX-XER 2008 test collection. While the proposed techniques were designed for the ER task, experimental results for the list completion task are consistent. While more experimentation is needed to conclude that the proposed techniques perform well in general, we have shown how they improve effectiveness on the used test collection.

We also saw that it might be possible to apply and/or combine different approaches depending on the query in order to maximize effectiveness—e.g., by using our methods we achieve an xInfAP value of over 0.7 for 20% of the queries of the used test collection and the mean xInfAP can be further boosted by 27% only by selecting the appropriate approach for each given topic. We leave as future work the research question of automatically selecting appropriate approaches for each query (e.g., by estimating the expected number of relevant results). We also point out that initial steps toward this goal have been done in Vercoustre et al. (2009) by applying machine learning techniques to predict query difficulty.

In this paper and in related work (see Sect. 7) it is possible to notice that precision values are low overall. This indicates that the entity ranking research field is only at its beginning and needs more work focusing on high precision algorithms in order to provide the users with a satisfying search experience. It is worthwhile investigating how to automatically determine (e.g., by statistics about the number of pages in the sought category or by frequency of categories for pages) when the category information should be used as-is and when this should be further processed or even ignored. Also, more focused research on NLP based techniques should be performed to broaden or narrow the query specificity depending on prediction of effectiveness by means of query analysis and classification. Finally, search effectiveness of the XER task can be further improved by using available collections (e.g., Wikipedia) annotated with state of the art NLP tools or enriched with semantic information (see, e.g., Jordi Atserias et al. 2008; Schenkel et al. 2007). A current limitation of this work is that the described algorithms are designed for the Wikipedia setting and can not be directly applied to the Web at large. It will be focus of our future work to extend the proposed methods for the Web of Entities.

Acknowledgments This work is partially supported by the EU Large-scale Integrating Projects OKKAM[11] - Enabling a Web of Entities (contract no. ICT-215032), PHAROS[12] (IST contract no. 045035), and LivingKnowledge[13] (contract no. 231126).

References

Adler, B. T., & de Alfaro, L. (2007). A content-driven reputation system for the wikipedia. In *WWW '07: Proceedings of the 16th international conference on World Wide Web* (pp. 261–270). New York, NY, USA: ACM.

Alias-i. (2008). LingPipe named entity tagger. Available at: http://www.alias-i.com/lingpipe/.

Allan, J., & Raghavan, H. (2002). Using part-of-speech patterns to reduce query ambiguity. In *SIGIR '02: Proceedings of the 25th annual international ACM SIGIR conference on research and development in information retrieval* (pp 307–314). New York, NY, USA: ACM.

Anick, P. G., & Tipirneni, S. (1999). The paraphrase search assistant: Terminological feedback for iterative information seeking. In *SIGIR '99: Proceedings of the 22nd annual international ACM SIGIR conference on research and development in information retrieval* (pp. 153–159). New York, NY, USA: ACM.

Bailey, P., de Vries, A. P., Craswell, N., & Soboroff, I. (2007). Overview of the TREC 2007 enterprise track. In E. M. Voorhees & L. P. Buckland (Eds.), *Proceedings of the sixteenth text REtrieval conference, TREC 2007, Gaithersburg, Maryland, USA, November 5–9, 2007*, volume Special Publication 500-274. National Institute of Standards and Technology (NIST).

Bast, H., Chitea, A., Suchanek, F., & Weber, I. (2007). ESTER: efficient search on text, entities, and relations. In *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on research and development in information retrieval* (pp. 671–678). New York, NY, USA: ACM.

Bhogal, J., Macfarlane, A., & Smith, P. (2007). A review of ontology based query expansion. *Information Processing & Management 43*(4), 866–886.

Bouquet, P., Halpin, H., Stoermer, H., & Tummarello, G. (Eds.). (2008) *Proceedings of the 1st international workshop on Identity and reference on the Semantic Web (IRSW2008) at the 5th European Semantic Web Conference (ESWC 2008), Tenerife, Spain, June 2, 2008*, CEUR workshop proceedings. CEUR-WS.org.

Bouquet, P., Stoermer, H., & Bazzanella, B. (2008). An Entity Name System (ENS) for the Semantic Web. In S. Bechhofer, M. Hauswirth, J. Hoffmann, & M. Koubarakis (Eds.), *The semantic web: Research and applications, 5th European Semantic Web Conference, ESWC 2008, Tenerife, Canary Islands, Spain, June 1–5, 2008, Proceedings*, volume 5021 of *Lecture notes in computer science* (pp. 258–272). New York: Springer.

Bouquet, P., Stoermer, H., Tummarello, G., & Halpin, H. (Eds.). (2007). *Proceedings of the WWW2007 workshop I^3: Identity, identifiers, identification, entity-centric approaches to information and knowledge management on the Web, Banff, Canada, May 8, 2007*, volume 249 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Broder, A. (2002). A taxonomy of web search. *SIGIR Forum, 36*(2), 3–10.

Carmel, D., Yom-Tov, E., & Soboroff, I. (2005). SIGIR workshop report: Predicting query difficulty—Methods and applications. *SIGIR Forum, 39*(2), 25–28.

Cheng, T., & Chang, K. C.-C. (2007). Entity search engine: Towards Agile best-effort information integration over the web. In *CIDR 2007, Third Biennial conference on innovative data systems research, Asilomar, CA, USA, January 7–10, 2007, Online Proceedings* (pp. 108–113). http://www.crdrdb.org.

Cheng, T., Yan, X., & Chang, K. C.-C. (2007). EntityRank: Searching entities directly and holistically. In *VLDB '07: Proceedings of the 33rd international conference on very large data bases* (pp. 387–398). VLDB Endowment.

[11] http://www.fp7.okkam.org/.

[12] http://www.pharos-audiovisual-search.eu/.

[13] http://www.livingknowledge-project.eu/.

Chirita, P. A., Firan, C. S., & Nejdl, W. (2007) Personalized query expansion for the Web. In *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on research and development in information retrieval* (pp. 7–14). New York, NY, USA: ACM.

Demartini, G., Firan, C. S., Iofciu, T., Krestel, R., & Nejdl, W. (2008). A model for ranking entities and its application to Wikipedia. In *LA-WEB* '08: *Proceedings of the 2008 latin American web conference* (pp. 29–38). Washington, DC, USA: IEEE Computer Society.

Demartini, G., Firan, C. S., Iofciu, T., & Nejdl, W. (2008). Semantically enhanced entity ranking. In *WISE '08: Proceedings of the 9th international conference on web information systems engineering* (pp. 176–188). Berlin, Heidelberg: Springer.

Denoyer, L., & Gallinari, P. (2006). The Wikipedia XML corpus. *SIGIR Forum, 40*(1), 64–69.

Fellbaum, C. (1998). *WordNet: An electronic lexical database (language, speech, and communication).* Cambridge: The MIT Press.

Heath, T., & Motta, E. (2008). Revyu: Linking reviews and ratings into the Web of data. *Journal of Web Semantics, 6*(4), 266–273.

Hsu, M.-H., Tsai, M.-F., & Chen, H.-H. (2006). Query expansion with ConceptNet and WordNet: An intrinsic comparison. In H. T. Ng, M.-K. Leong, M.-Y. Kan, & D. Ji (Eds), *Information retrieval technology, third Asia information retrieval symposium, AIRS 2006, Singapore, October 16–18, 2006, Proceedings*, volume 4182 of *Lecture notes in computer ccience* (pp. 1–13). New York: Springer.

Jordi Atserias, M. C., Zaragoza, H., & Attardi, G. (2008). Semantically annotated snapshot of the English Wikipedia. In N. Calzolari, K. Choukri, B. Maegaard, J. Mariani, J. Odjik, S. Piperidis, & D. Tapias (Eds), *Proceedings of the sixth international language resources and evaluation (LREC'08)*, Marrakech, Morocco, May 2008. European Language Resources Association (ELRA). http://www.lrec-conf.org/proceedings/lrec2008/.

Palpanas, T., Chaudhry, J., Andritsos, P., & Velegrakis, Y. (2008). Entity data management in OKKAM. In *DEXA '08: Proceedings of the 2008 19th international conference on database and expert systems application* (pp. 729–733). Washington, DC, USA: IEEE Computer Society.

Pehcevski, J., Vercoustre, A.-M., & Thom, J. A. (2008). Exploiting locality of Wikipedia links in entity ranking. In C. Macdonald, I. Ounis, V. Plachouras, I. Ruthven, & R. W. White (Eds), *Advances in information retrieval, 30th European conference on IR research, ECIR 2008, Glasgow, UK, March 30–April 3, 2008. Proceedings*, volume 4956 of *Lecture notes in computer science* (pp. 258–269). New York: Springer.

Raimond, Y., Sutton, C., & Sandler, M. (2008). Automatic interlinking of music datasets on the semantic web. In *Linked Data on the Web (LDOW2008).*

Rode, H., Hiemstra, D., Vries, A., & Serdyukov, P. (2009). Efficient XML and entity retrieval with PF/Tijah: CWI and University of Twente at INEX'08. In *Advances in focused retrieval: 7th international workshop of the initiative for the evaluation of XML retrieval, INEX 2008, Dagstuhl Castle, Germany, December 15–18, 2008. Revised and selected papers* (pp. 207–217). Berlin, Heidelberg: Springer.

Rölleke, T., Tsikrika, T., & Kazai, G. (2006). A general matrix framework for modelling information retrieval. *Information Processing & Management, 42*(1), 4–30.

Schenkel, R., Suchanek, F. M., & Kasneci, G. (2007). YAWN: A semantically annotated Wikipedia XML corpus. In A. Kemper, H. Schöning, T. Rose, M. Jarke, T. Seidl, C. Quix, & C. Brochhaus (Eds), *Datenbanksysteme in Business, Technologie und Web (BTW 2007), 12. Fachtagung des GI-Fachbereichs "Datenbanken und Informationssysteme" (DBIS), Proceedings, 7–9 März 2007, Aachen, Germany*, volume 103 of *LNI* (pp. 277–291). GI.

Semeraro, G., Degemmis, M., Lops, P., & Basile, P. (2007). Combining learning and word sense disambiguation for intelligent user profiling. In *IJCAI'07: Proceedings of the 20th international joint conference on artifical intelligence* (pp. 2856–2861) San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Suchanek, F. M., Kasneci, G., & Weikum, G. (2007). Yago: a core of semantic knowledge. In C. L. Williamson, M. E. Zurko, P. F. Patel-Schneider, & P. J. Shenoy (Eds.), *Proceedings of the 16th international conference on World Wide Web, WWW 2007, Banff, Alberta, Canada, May 8–12, 2007* (pp. 697–706). New York: ACM.

Tsikrika, T., Serdyukov, P., Rode, H., Westerveld, T., Aly, R., Hiemstra, D., & Vries, A. P. (2008). Structured document retrieval, multimedia retrieval, and entity ranking using PF/Tijah. In *Focused access to XML documents: 6th international workshop of the initiative for the evaluation of XML retrieval, INEX 2007 Dagstuhl Castle, Germany, December 17–19, 2007. Selected papers* (pp. 306–320). Berlin, Heidelberg: Springer.

Vercoustre, A.-M., Pehcevski, J., & Naumovski, V. (2009). Topic difficulty prediction in entity ranking. In *Advances in focused retrieval: 7th international workshop of the initiative for the evaluation of XML* *retrieval, INEX 2008, Dagstuhl Castle, Germany, December 15–18, 2008. Revised and selected papers* (pp. 280–291). Berlin, Heidelberg: Springer.

Voorhees, E. M. (1993). On expanding query vectors with lexically related words. In *TREC* (pp. 223–232).

Voorhees, E. M. (1994). Query expansion using lexical-semantic relations. In *SIGIR '94: Proceedings of the 17th annual international ACM SIGIR conference on research and development in information retrieval* (pp. 61–69). New York, NY, USA: Springer.

Vries, A. P., Vercoustre, A.-M., Thom, J. A., Craswell, N., & Lalmas, M. (2007). *0verview of the INEX 2007 entity ranking track. In Focused access to XML documents: 6th international workshop of the initiative for the evaluation of XML retrieval, INEX 2007 Dagstuhl Castle, Germany, December 17–19, 2007. Selected Papers* (pp. 245–251). Berlin, Heidelberg: Springer-Verlag.

Webber, W., Moffat, A., Zobel, J., & Sakai, T. (2008). Precision-at-ten considered redundant. In *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on research and development in information retrieval* (pp. 695–696). New York, NY, USA: ACM.

Yilmaz, E., & Aslam, J. A. (2006). Estimating average precision with incomplete and imperfect judgments. In *CIKM '06: Proceedings of the 15th ACM international conference on Information and knowledge management* (pp. 102–111). New York, NY, USA: ACM.

Yilmaz, E., Kanoulas, E., & Aslam, J. A. (2008). A simple and efficient sampling method for estimating AP and NDCG. In *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 603–610), New York, NY, USA: ACM.

Zaragoza, H., Rode, H., Mika, P., Atserias, J., Ciaramita, M., & Attardi, G. (2007) Ranking very many typed entities on Wikipedia. In *CIKM '07: Proceedings of the sixteenth ACM conference on conference on information and knowledge management* (pp. 1015–1018). New York, NY, USA: ACM.

Zhu, J., de Vries, A. P., Demartini, G., & Iofciu, T. (2008). Relation retrieval for entities and experts. In *Future challenges in expertise retrieval (fCHER 2008), SIGIR 2008 Workshop.*

Zirn, C., Nastase, V., & Strube, M. (2008). Distinguishing between Instances and Classes in the Wikipedia taxonomy. In S. Bechhofer, M. Hauswirth, J. Hoffmann, & M. Koubarakis (Eds), *The semantic web: Research and applications, 5th European semantic web conference, ESWC 2008, Tenerife, Canary Islands, Spain, June 1–5, 2008, Proceedings*, volume 5021 of *Lecture notes in computer science* (pp. 376–387). New York: Springer.